CLAIMS

What is claimed is:

1. An apparatus comprising:

a peripheral bus coupled to a peripheral unit to transfer peripheral information including a command message specifying a peripheral operation; and

a processing slice coupled to the peripheral bus to execute a plurality of threads, the plurality of threads including a first thread sending the command message to the peripheral unit.

2. The apparatus of claim 1 wherein the peripheral unit is one of an input device and an output device.

3. The apparatus of claim 1 wherein the peripheral operation is one of an input operation and an output operation.

4. The apparatus of claim 1 wherein the command messages includes at least one of a message content, a peripheral address identifying the peripheral unit, and a command code specifying the peripheral operation.

5. The apparatus of claim 1 wherein the peripheral information includes a response message sent from the peripheral unit to the processing slice, the response message indicating the peripheral operation is completed.

6. The apparatus of claim 5 wherein the response message includes at least one of a thread identifier identifying the first thread, an operation result of the peripheral operation, a data register address specifying a data register in the processing slice to store the operation result, and a length indicator indicating length of the response message.

7. The apparatus of claim 6 wherein the peripheral bus comprises:

a bi-directional bus to transfer the command message from the processing slice to the peripheral unit and the response message from peripheral unit to the processing slice.

8. The apparatus of claim 1 wherein the processing slice disables the first thread after sending the command message if the command message is a wait instruction.

9. The apparatus of claim 1 wherein the first thread continues to execute after sending the command message if the command message is a non-wait instruction.

10. The apparatus of claim 8 wherein the processing slice enables the first thread after receiving the response message from the peripheral unit if the first thread was disabled.

11. The apparatus of claim 1 wherein the processing slice comprises:

an instruction processing unit to process instructions fetched from a program memory; and

a thread control unit coupled to the instruction processing unit to manage initiating and termination of at least one of the plurality of threads.

12. The apparatus of claim 11 wherein the processing slice further comprises:

a memory access unit coupled to the instruction processing unit to provide access to one of a plurality of data memories via a data memory switch, the memory access unit having a plurality of data base registers, each of the data base registers corresponding to each of the threads; and

a functional unit coupled to the instruction processing unit to perform an operation specified in one of the instructions; and

a register file coupled to the instruction processing unit and the peripheral unit having a plurality of data registers, each of the data registers corresponding to each of the threads.

13. The apparatus of claim 12 wherein the instruction processing unit comprises:

an instruction fetch unit to fetch the instructions from the program memory using a plurality of program counters, each program counter corresponding to each of the threads;

an instruction buffer coupled to the instruction fetch unit to hold the fetched instructions; and

an instruction decoder and dispatcher coupled to the instruction buffer to decode the instructions and dispatch the decoded instructions to one of the memory access unit, the functional unit, and the peripheral unit;

wherein the instructions are executed concurrently in a clock cycle.

14. A method comprising:

transferring peripheral information to a peripheral unit via a peripheral bus, the peripheral information including a command message specifying a peripheral operation; and

executing a plurality of threads by a processing slice, the plurality of threads including a first thread sending the command message to the peripheral unit.

15. The method of claim 14 wherein the peripheral unit is one of an input device and an output device.

16. The method of claim 14 wherein the peripheral operation is one of an input operation and an output operation.

17. The method of claim 14 wherein the command messages includes at least one of a message content, a peripheral address identifying the peripheral unit, and a command code specifying the peripheral operation.

18. The method of claim 14 wherein the peripheral information includes a response message sent from the peripheral unit to the processing slice, the response message indicating the peripheral operation is completed.

19. The method of claim 18 wherein the response message includes at least one of a thread identifier identifying the first thread, an operation result of the peripheral operation, a data register address specifying a data register in the processing slice to store the operation result, and a length indicator indicating length of the response message.

20. The method of claim 19 wherein transferring the peripheral information comprises:

transferring the command message from the processing slice to the peripheral unit and the response message from peripheral unit to the processing slice via a bi-directional bus.

21. The method of claim 14 wherein executing the plurality of threads comprises disabling the first thread after sending the command message if the command message is a wait instruction.

22. The method of claim 14 wherein executing the plurality of threads comprises continuing executing the first thread after sending the command message if the command message is a non-wait instruction.

23. The method of claim 21 wherein executing the plurality of threads comprises enabling the first thread after receiving the response message from the peripheral unit if the first thread was disabled.

24. The method of claim 14 wherein executing the plurality of threads comprises:

processing instructions fetched from a program memory by an instruction processing unit;

managing initiating and termination of at least one of the plurality of threads b a thread control unit.

25. The method of claim 24 wherein executing the plurality of threads further comprises:

accessing to one of a plurality of data memories by a memory access unit via a data memory switch, the memory access unit having a plurality of data base registers, each of the data base registers corresponding to each of the threads;

performing an operation specified in one of the instructions by a functional unit; and

storing data in a register file having a plurality of data registers, each of the data registers corresponding to each of the threads.

26. The method of claim 24 wherein processing instructions comprises:

fetching the instructions from the program memory using a plurality of program counters by an instruction fetch unit, each program counter corresponding to each of the threads;

holding the fetched instructions in an instruction buffer;

decoding the instructions and dispatching the decoded instructions by an instruction decoder and dispatcher to one of the memory access unit, the functional unit, and the peripheral unit; and

executing the instructions concurrently in a clock cycle.

27. A processing system comprising:

a plurality of banks of data memory;

a data memory switch coupled to the banks to data memory;

a program memory to store a program;

a peripheral bus coupled to a peripheral unit to transfer peripheral information including a command message specifying a peripheral operation; and

a processing slice coupled to the peripheral bus to execute a plurality of threads, the plurality of threads including a first thread sending the command message to the peripheral unit.

28. The processing system of claim 27 wherein the peripheral unit is one of an input device and an output device.

29. The processing system of claim 27 wherein the peripheral operation is one of an input operation and an output operation.

30. The processing system of claim 27 wherein the command messages includes at least one of a message content, a peripheral address identifying the peripheral unit, and a command code specifying the peripheral operation.

31. The processing system of claim 27 wherein the peripheral information includes a response message sent from the peripheral unit to the processing slice, the response message indicating the peripheral operation is completed.

32. The processing system of claim 31 wherein the response message includes at least one of a thread identifier identifying the first thread, an operation result of the peripheral operation, a data register address specifying a data register in the processing slice to store the operation result, and a length indicator indicating length of the response message.

33. The processing system of claim 32 wherein the peripheral bus comprises:

a bi-directional bus to transfer the command message from the processing slice to the peripheral unit and the response message from peripheral unit to the processing slice.

34. The processing system of claim 27 wherein the processing slice disables the first thread after sending the command message if the command message is a wait instruction.

35. The processing system of claim 27 wherein the first thread continues to execute after sending the command message if the command message is a non-wait instruction.

36. The processing system of claim 34 wherein the processing slice enables the first thread after receiving the response message from the peripheral unit if the first thread was disabled.

37. The processing system of claim 27 wherein the processing slice comprises:

an instruction processing unit to process instructions fetched from a program memory; and

a thread control unit coupled to the instruction processing unit to manage initiating and termination of at least one of the plurality of threads.

38. The processing system of claim 37 wherein the processing slice further comprises:

a memory access unit coupled to the instruction processing unit to provide access to one of a plurality of data memories via a data memory switch, the memory access unit having a plurality of data base registers, each of the data base registers corresponding to each of the threads;

a functional unit coupled to the instruction processing unit to perform an operation specified in one of the instructions; and

a register file coupled to the instruction processing unit and the peripheral unit having a plurality of data registers, each of the data registers corresponding to each of the threads.

39. The processing system of claim 38 wherein the instruction processing unit comprises:

an instruction fetch unit to fetch the instructions from the program memory using a plurality of program counters, each program counter corresponding to each of the threads;

an instruction buffer coupled to the instruction fetch unit to hold the fetched instructions; and

an instruction decoder and dispatcher coupled to the instruction buffer to decode the instructions and dispatch the decoded instructions to one of the memory access unit, the functional unit, and the peripheral unit;

wherein the instructions are executed concurrently in a clock cycle.

40. A processing system comprising:

a plurality of multi-thread processors;

a plurality of peripheral units;

a peripheral bus coupled to the peripheral units to transfer peripheral information between the multi-thread processors and the peripheral units, the peripheral information including a command message sent from one of the multi-thread processors to one of the peripheral units by a thread executing a message instruction.

41. A processing system comprising:

a multi-thread processor having program base registers and data base registers;

at least one peripheral units;

a peripheral bus coupled to the at least one peripheral unit to transfer peripheral information between the multi-thread processor and the at least one peripheral unit, the peripheral information including a command message sent from one of the multi-thread processors to one of the peripheral units by a thread executing a message instruction.